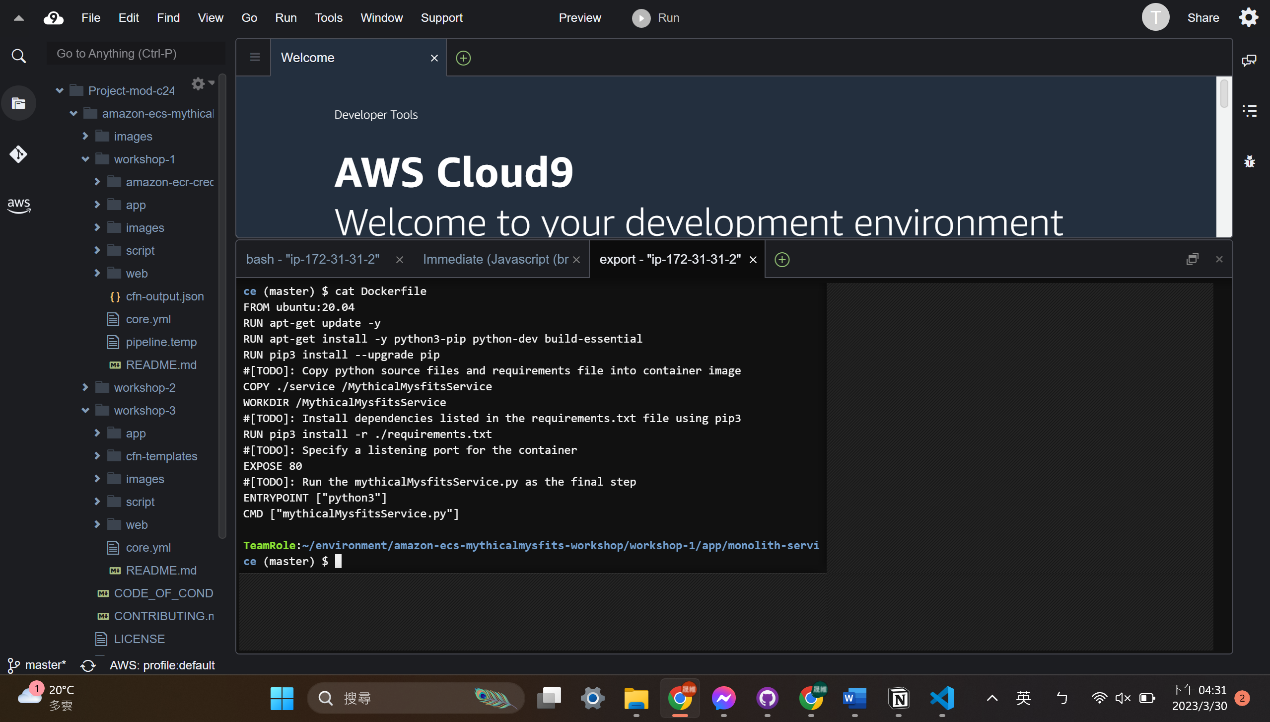
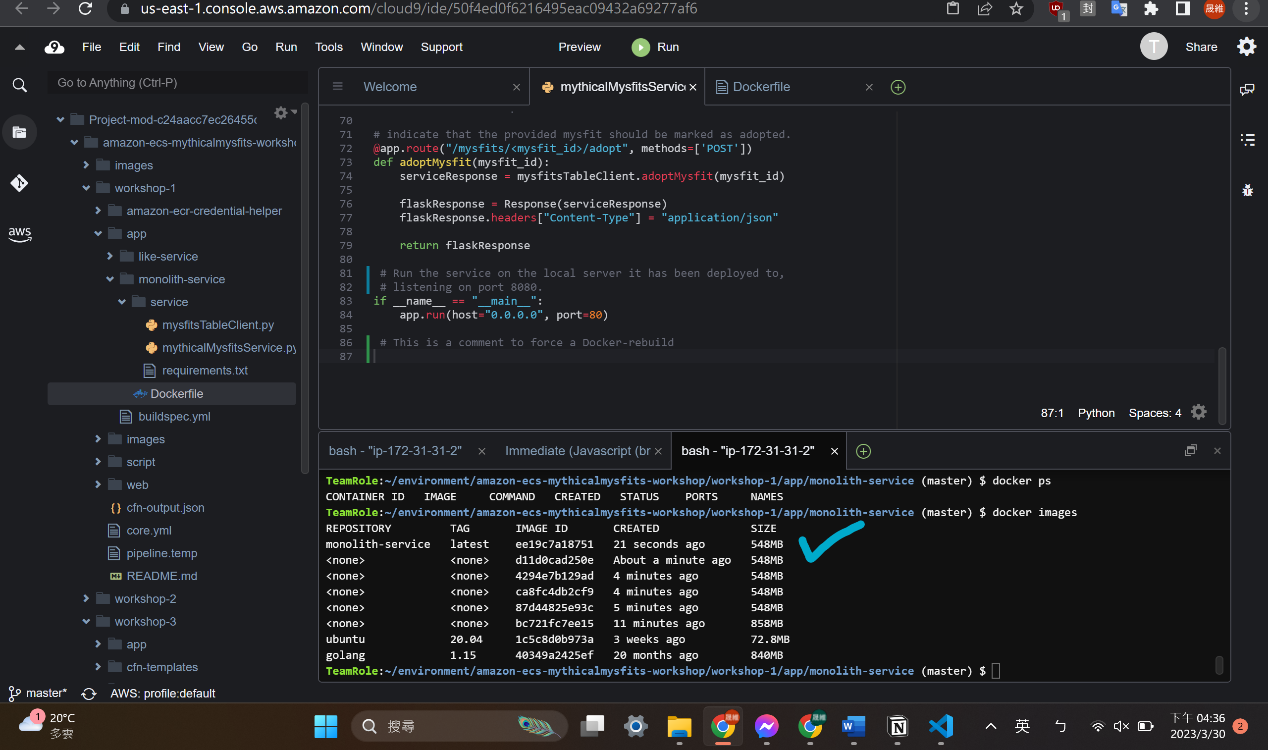
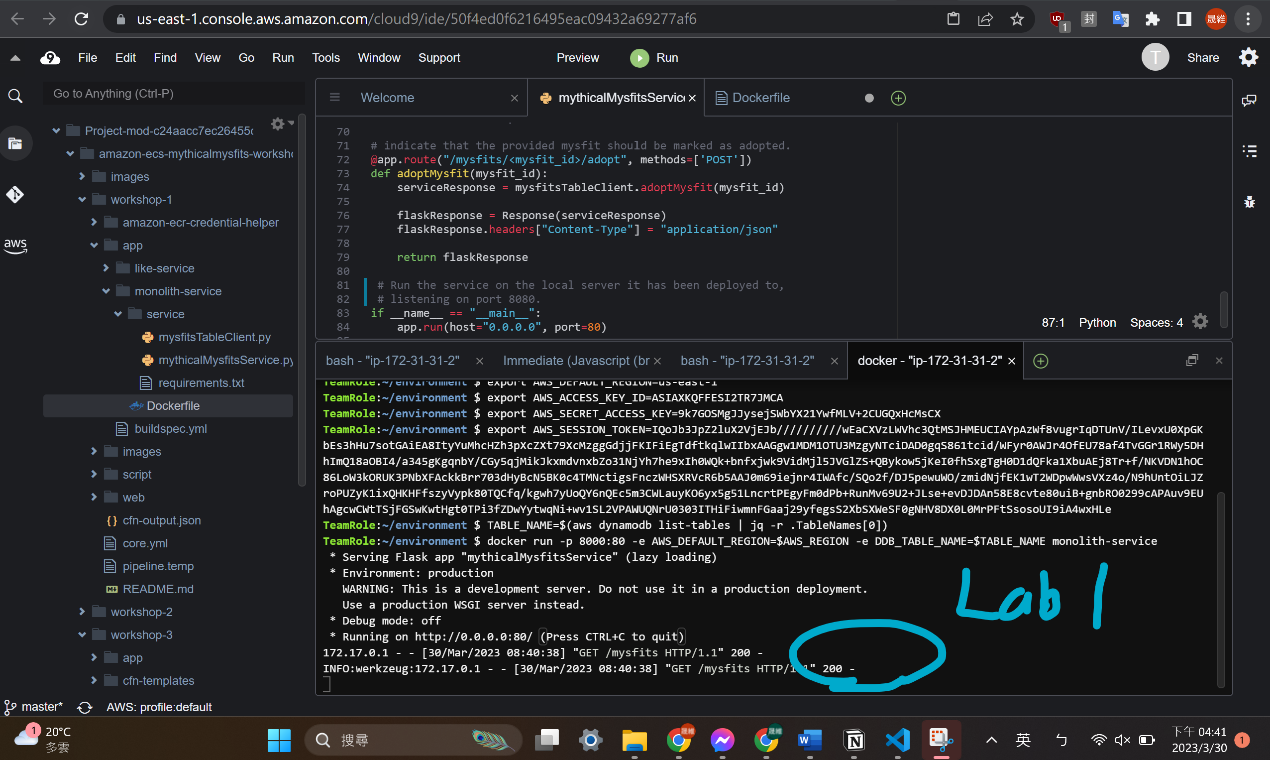
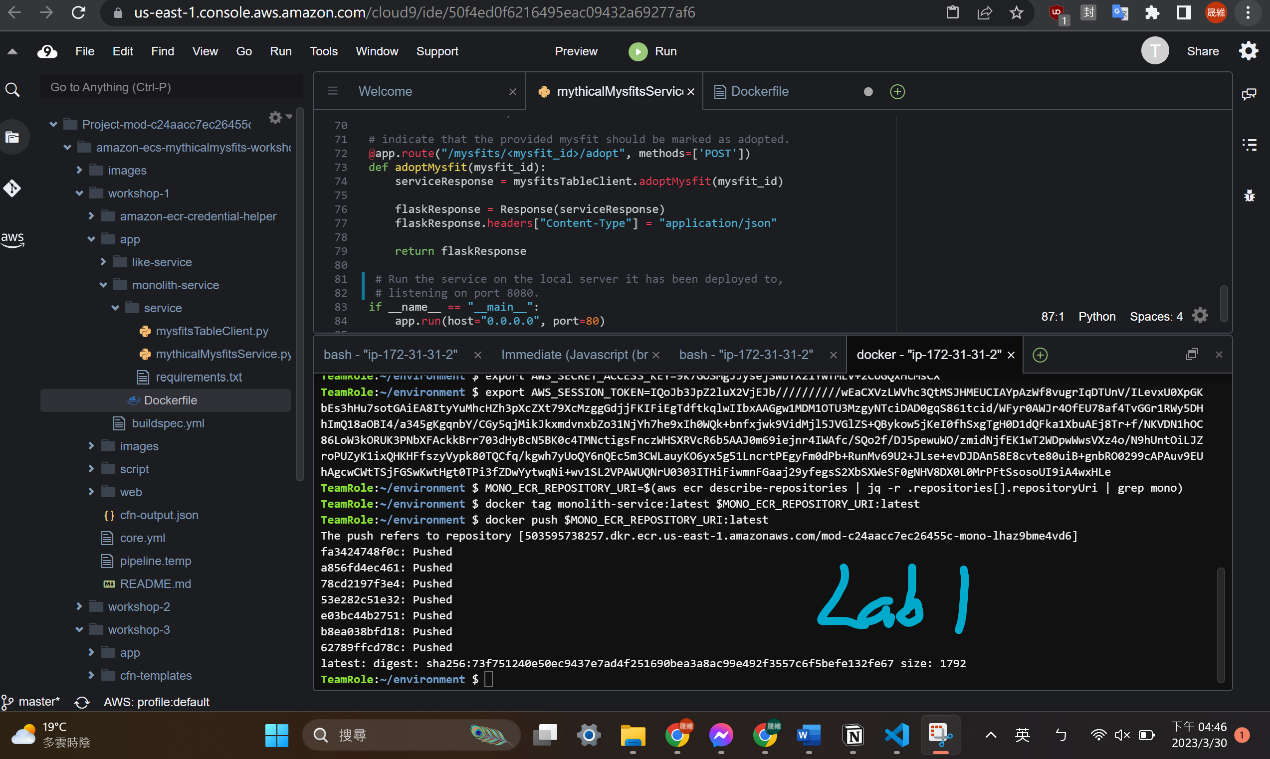
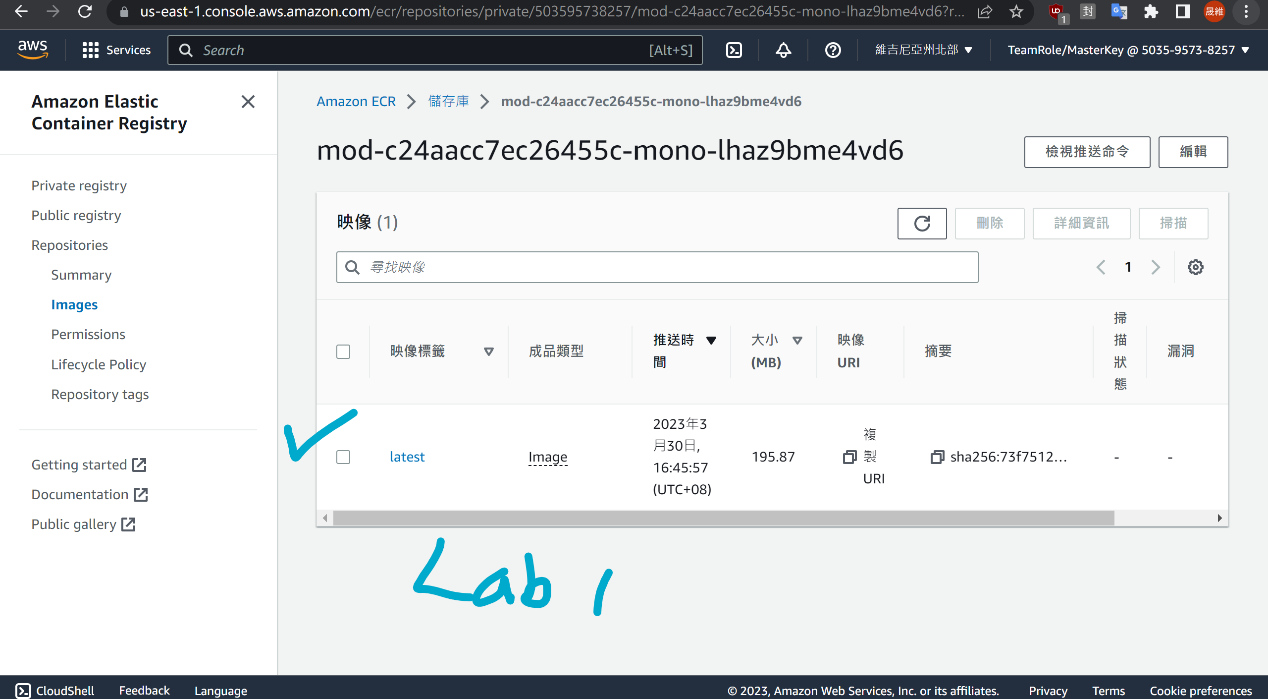
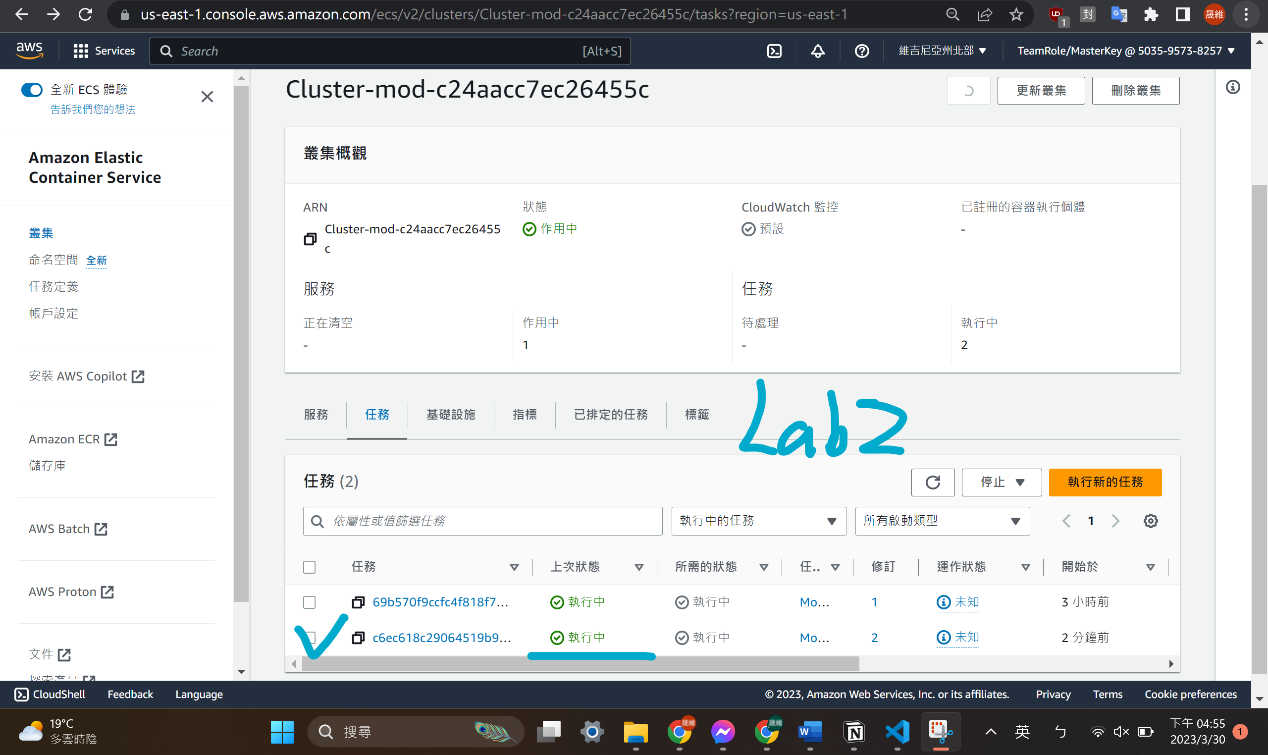
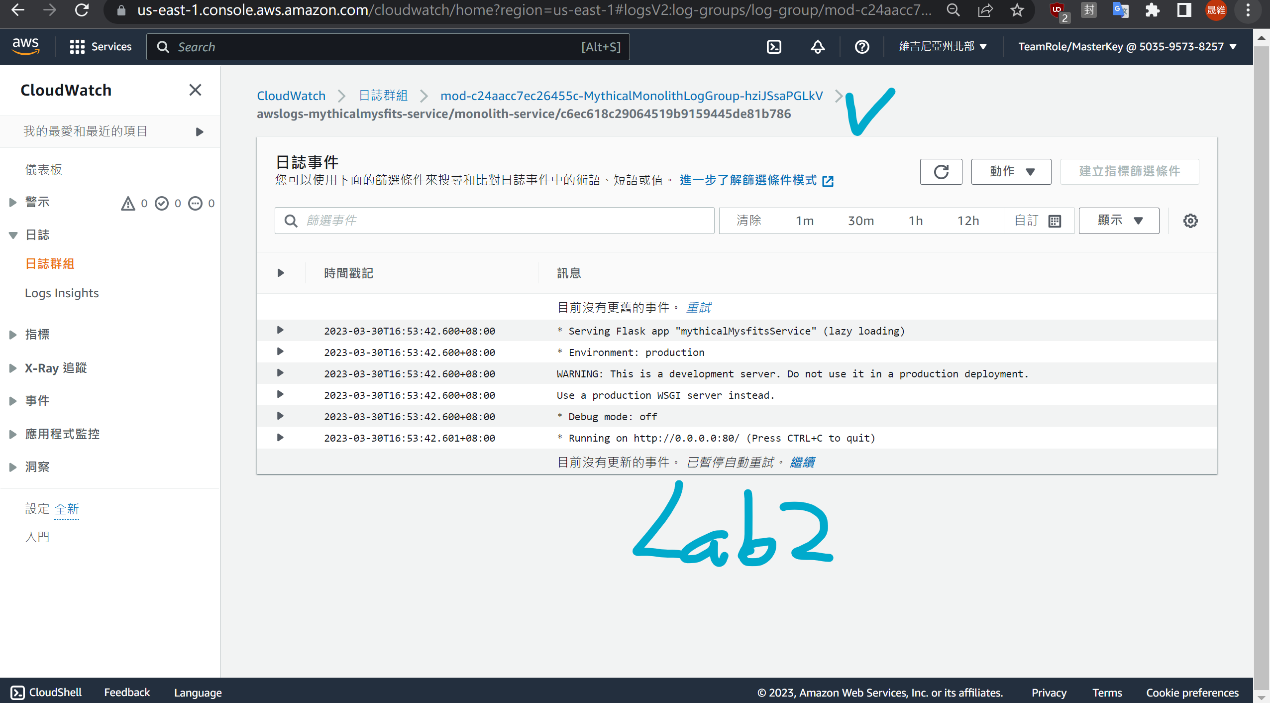
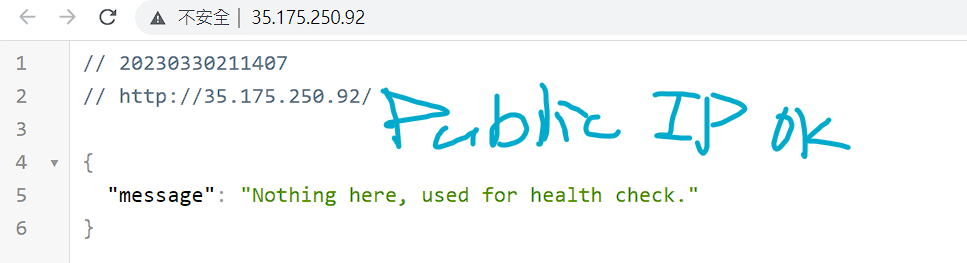
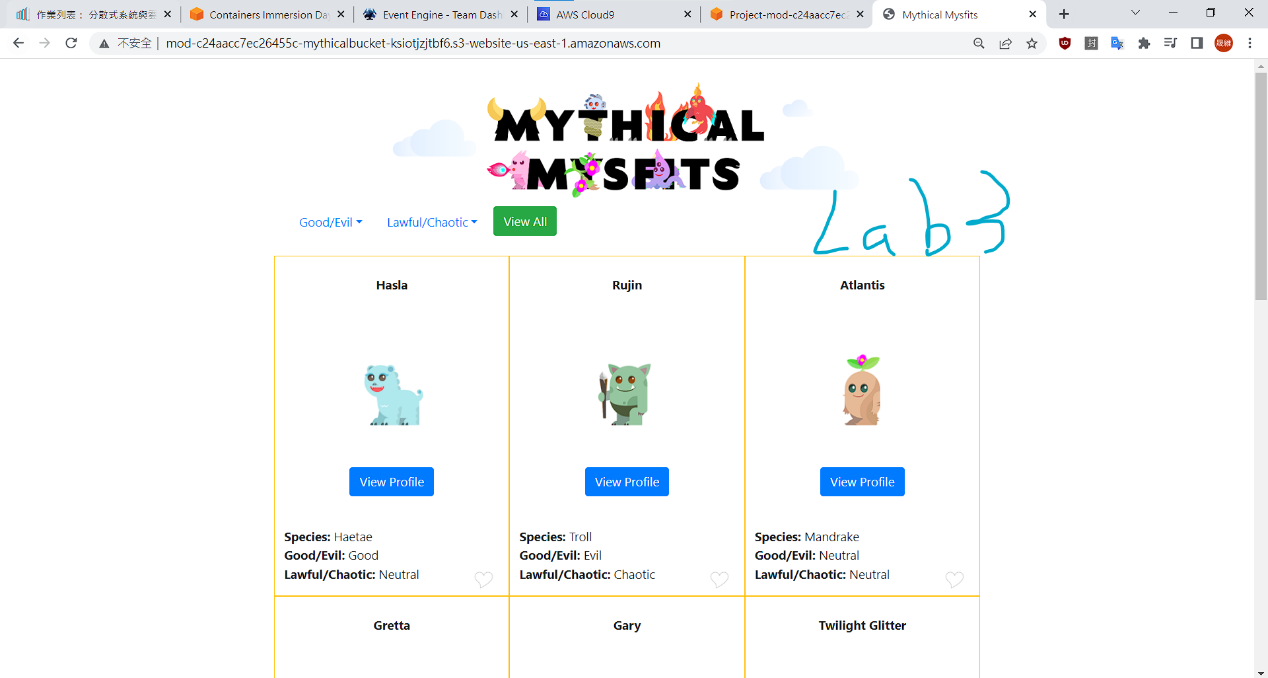
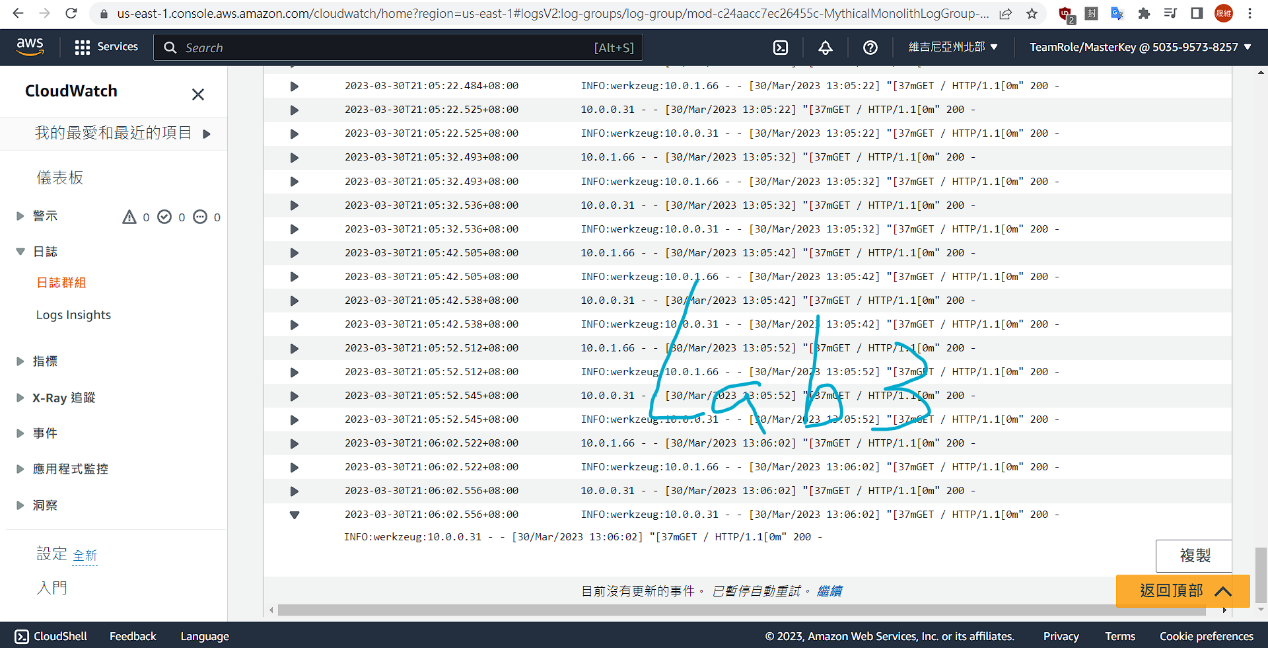
**Lab 1. Containerize the Mythical Mysfits monolith**

* 目的: 轉寫一個希臘神話的網頁，製作成 docker image 發布到 ECR 上並使用 ECS 部屬網站。最後將功能切分，以 loadbalancer 調節 api 流量。
* 撰寫 docker file 並建立web app image
  + 
  + 
* Docker run 本地的容器
  + 
* 將 image push 到 [**Amazon Elastic Container Registry**](https://us-east-1.console.aws.amazon.com/ecr/get-started?region=us-east-1) 上
  + 
  + 

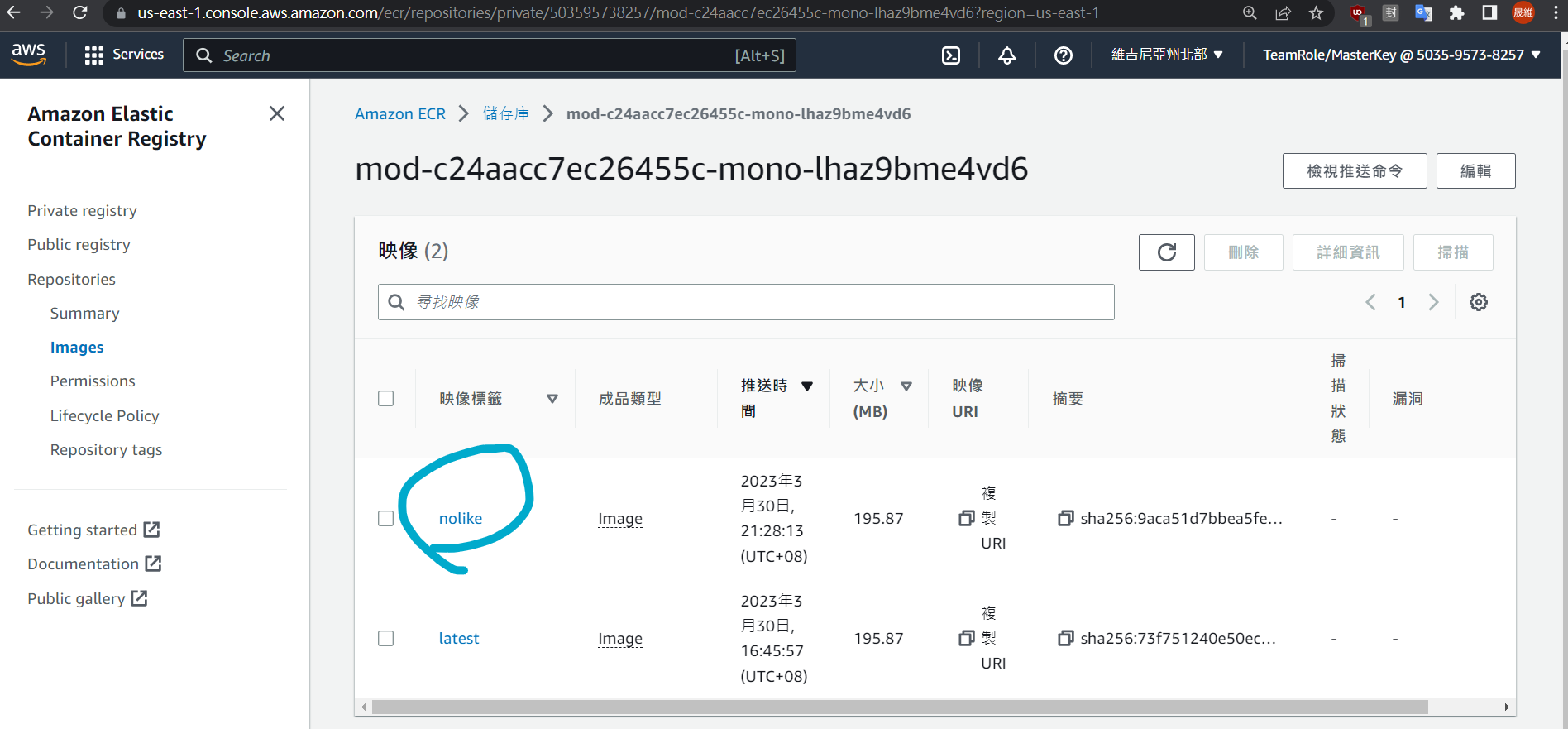
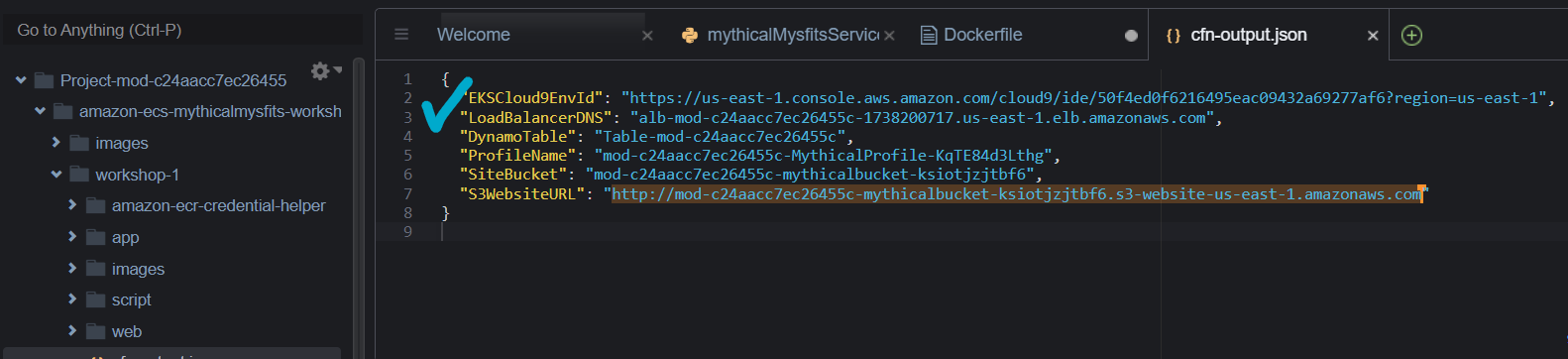
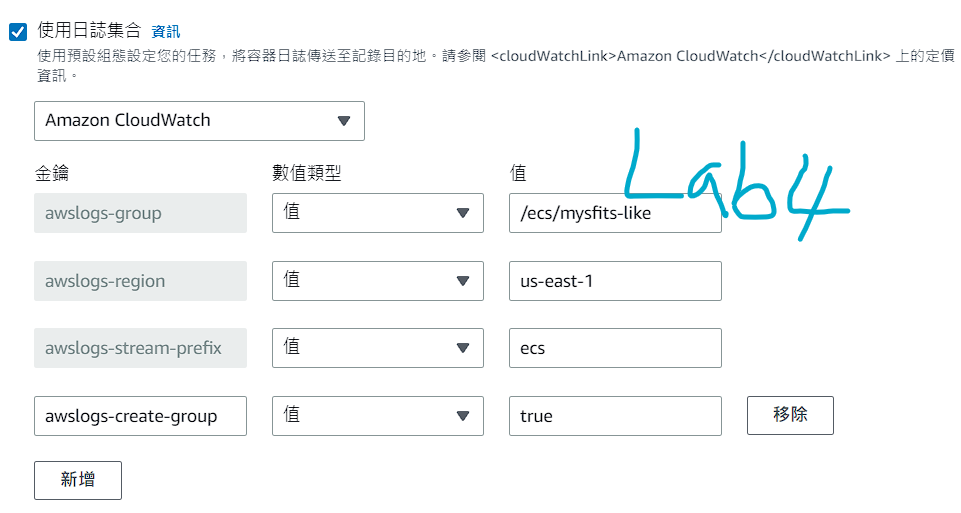
**Lab 2. Deploy the container using AWS Fargate**

* 把Task建立在Amazon Elastic Container Service (ECS)上
  + 
* 可以從外部IP的HTTP連上剛剛部屬好的task，透過API取得希臘神的資料。CloudWatch Logs dashboard 的 log 也能順利顯示部屬和API造訪資訊。
  + 
  + 

**Lab 3. Scale the adoption platform monolith with an ALB and an ECS Service**

* 這個任務是要套用Application Load Balancer (ALB)
* 先確認 NGigx可用
  + 
* 確認 Public IP有成功對外開放
  + 
* 更新 ECS cluster 的 service，現在是第二版。並且小網站的UI更新上來了。也可以對她 like。
  + 
  + 

**Lab 4. Incrementally build and deploy more microservices with AWS Fargate**

* 這個lab的目標是要將不容的服務分流，如 *'/adopt' and '/like'*，和資料庫的存取劃分為不同服務。
* 將“like”功能移動到單獨的服務中。
  + 
* 另外 build 和 push 一個 image “nolike”。
  + 
* 新增一個全新的 Task 結果如下
  + 
* 詳細步驟 (1) 設定 Container Details 以及 ALB DNS name，可以在 code 的 config 檔中看到
  + 
  + 
  + 詳細步驟 (2)
  + 設定每個 bucket (?) 的硬體環境，如CPU和RAM，權限等等。
  + 
* 詳細步驟 (3)
* 紀錄 Logs
  + 
* 詳細步驟 (4)
* 設定已制定好的 load balancer，for listener, select Use an existing listener and choose 80:\HTTP from the drop-down.
  + 
  + 
* 最後確認小網站正常運作，一樣可以點愛心。
  + 